PAPER

Quality Analysis of Discretization Methods for Estimation of Distribution Algorithms

Chao-Hong CHEN[†a)], *Nonmember and* Ying-ping CHEN[†b)], *Member*

SUMMARY Estimation of distribution algorithms (EDAs), since they were introduced, have been successfully used to solve discrete optimization problems and hence proven to be an effective methodology for discrete optimization. To enhance the applicability of EDAs, researchers started to integrate EDAs with discretization methods such that the EDAs designed for discrete variables can be made capable of solving continuous optimization problems. In order to further our understandings of the collaboration between EDAs and discretization methods, in this paper, we propose a quality measure of discretization methods for EDAs. We then utilize the proposed quality measure to analyze three discretization methods: fixed-width histogram (FWH), fixed-height histogram (FHH), and greedy random split (GRS). Analytical measurements are obtained for FHH and FWH, and sampling measurements are conducted for FHH, FWH, and GRS. Furthermore, we integrate Bayesian optimization algorithm (BOA), a representative EDA, with the three discretization methods to conduct experiments and to observe the performance difference. A good agreement is reached between the discretization quality measurements and the numerical optimization results. The empirical results show that the proposed quality measure can be considered as an indicator of the suitability for a discretization method to work with EDAs.

key words: *quality analysis, discretization distortion, fixed-width histogram, fixed-height histogram, greedy random split, estimation of distribution algorithm, Bayesian optimization algorithm*

1. Introduction

Genetic algorithms (GAs) [11], [15] are methodologies inspired by Darwinian evolution and widely applied to real-world problems. In genetic algorithms, good individuals are selected from the current population to generate the next population by using recombination and mutation operators, mimicking the biological genetic operations. According to the theory of design decomposition [12], the key components to the GA success include identifying, reproducing, and exchanging the structure of the solutions. Recombination, one of the main GA operator, mixes the promising sub-solutions, called building blocks (BBs), and creates the solutions. Because the recombination operator mixes promising sub-solutions and creates new solutions, genetic algorithms work well on the problems which can be implicitly or explicitly decomposed into sub-problems.

In order to appropriately mix genes, the evolutionary algorithms based on utilizing probabilistic models were proposed and developed [19], [26]. In these schemes, instead of using regular recombination and mutation operators, the offspring population is created according to the estimated probabilistic model of the selected individuals of the current population. The probabilistic model is expected to reflect the problem structure, and better performance can be achieved via exploring the relationship between genes. These evolutionary algorithms are called *estimation of distribution algorithms* (EDAs). In EDAs, decision variables are often coded with discrete codes, such as binary codes. To enhance the applicability of EDAs over continuous domains, direct attempts to modify the type of decision variables have been made, including continuous population-based incremental learning with Gaussian distribution [31], real-coded variant of population-based incremental learning with interval updating [32], Bayesian evolutionary algorithms for continuous function optimization [33], real-coded extended compact genetic algorithm based on mixtures of models [18], and the real-coded Bayesian optimization algorithm [1].

Instead of modifying the infrastructure of the algorithm, such as the type of decision variables or the global program flow, as a more general, component-wise approach, discretization methods are employed to cooperate with EDAs [6], [7], [28], [35]. Discretization methods enable EDAs designed for discrete variables to solve continuous optimization problems without the need of altering algorithmic structures. For further information about these two approaches which enable EDAs to handle continuous variables, the reader may refer to [4], [5], [16]. In order to further our understandings of the collaboration between EDAs and discretization methods, as the first step, we wish to quantify the discretization quality and identify the suitability of discretization methods to work with EDAs. Particularly, in this paper, we firstly propose a quality measure of discretization methods and then utilize the proposed quality measure to analyze three discretization methods, fixed-width histogram (FWH), fixed-height histogram (FHH), and greedy random split (GRS). To observe the performance difference of these discretization methods in action, we integrate Bayesian optimization algorithm (BOA) [27] with the three methods to conduct numerical experiments. A good agreement between the measurements of discretization quality and the numerical results on test functions is obtained. Such an outcome indicates that the proposed quality measure indeed reflects the suitability for a discretization method to work with EDAs.

In the next section, we give a background of this study, including the introduction of EDAs and the three aforemen-

Manuscript received August 27, 2013.
Manuscript revised December 21, 2013.

[†] The authors are with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan.
a) E-mail: chchen@nclab.tw
b) E-mail: ypchen@nclab.tw

DOI: 10.1587/transinf.E97.D.1312

tioned discretization methods. Section 3 proposes the quality measure of discretization methods and analyzes the three methods by using the proposed measure. The numerical experiments of the three methods integrated with BOA on test functions are presented and discussed in Sect. 4, followed by the summary and conclusions in Sect. 5.

2. Background

In this section, we will give an overview of estimation of distribution algorithms. Then, three discretization methods, fixed-width histogram, fixed-height histogram, and greedy random split, are described.

2.1 Estimation of Distribution Algorithms

Estimation of distribution algorithms (EDAs) [19], [20], [23], [26] solve problems by building probabilistic models on promising solutions and generating offspring of the next generation from the model. EDAs replace the regular genetic operators, such as crossover and mutation, with the construction and sampling of probabilistic models. In EDAs, at each generation, the selection operator selects good individuals from the current population, and a probabilistic model is constructed based on these selected individuals. Then, a new population is generated by sampling the built model. An EDA scheme can be algorithmically outlined as

1. Initialize a population randomly.
2. Apply the selection operator on the population.
3. Build a probabilistic model from the selected individuals.
4. Generate a new population by sampling the model.
5. Stop if the termination criterion is satisfied.
6. Return to step 2.

According to [26], EDAs can be broadly categorized into three types. (1) No interaction: Each variable is modeled independently. These algorithms include the population-based incremental learning (PBIL) [2], the compact genetic algorithm (cGA) [14], and the univariate marginal distribution algorithm [23]. (2) Pairwise interaction: For these algorithms, pairwise interactions between variables are assumed, such as mutual-information-maximizing input clustering (MIMIC) [9], Baluja's dependency tree approach [3], and the bivariate marginal distribution algorithm (BMDA) [29]. (3) Multivariate interaction: These algorithms adopt models that can explore multivariate interactions, such as the extended compact genetic algorithm (ECGA) [13], the Bayesian optimization algorithm (BOA) [25], the estimation of Bayesian network algorithm (EBNA) [10], the factorized distribution algorithm (FDA) [22], the learning version of FDA (LFDA) [21] and the incremental Bayesian optimization algorithm (iBOA) [30]. With learning the dependencies among variables, the algorithms in this category can achieve better performance in certain sets of problems.

2.2 Discretization Methods

Discretization methods can be utilized to transform optimization problems from the continuous domain into the discrete domain and to provide an interface between problems and solvers in different domains. After being discretized, solution individuals in the continuous domain are encoded with certain discrete codes, such as binary strings or integer vectors. Then, EDAs originally designed for handling discrete decision variables can optimize transformed continuous problems. In this study, we consider only the discretization methods which divide the search space into intervals. The probability of search points in each interval are assumed to be uniformly distributed. A scheme to integrate discretization into EDAs can be outlined as

1. Initialize a population randomly.
2. **Conduct discretization to encode the population.**
3. Apply the selection operator on the population.
4. Build a probabilistic model from the selected individuals.
5. Generate a new population by sampling the model.
6. **Sample the coded intervals to create the continuous offspring.**
7. Stop if the termination criterion is satisfied.
8. Return to step 2.

In the remainder of this section, we will introduce two elementary discretization methods, fixed-width histogram (FWH) and fixed-height histogram (FHH) [17], [35], and a simple randomized discretization method, greedy random split (GRS).

2.2.1 Fixed-Width Histogram

In fixed-width histogram, each dimension is divided into several equal-width intervals, which are usually called *bins*. Each bin is assigned a discrete code or an integer. Each real-valued individual is coded with the code or integer assigned to the bin. If the search space for a dimension is $[L, U)$, a k-bin FWH will divide this interval into k bins with equal-width, $(U-L)/k$, and the range of the ith bin is

$$\left[L + \frac{(i-1)(U-L)}{k}, L + \frac{i(U-L)}{k}\right) ,$$

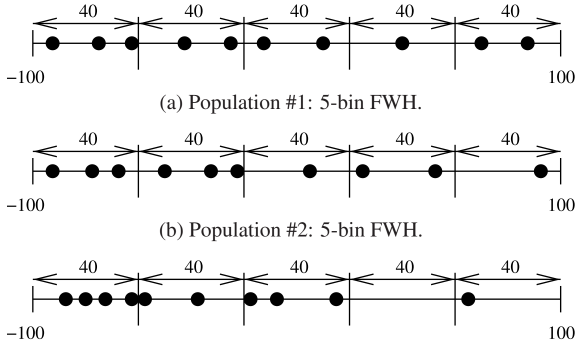
for $i = 1, \ldots, k$. Figure 1 shows three different variables discretized by a 5-bin FWH. The fixed width of each bin is 40.

2.2.2 Fixed-Height Histogram

Fixed-height histogram is similar to FWH, but instead of fixing the bin size, the "height" of each bin is fixed, i.e., each bin contains the same number of search points. If the points are concentrated in some regions, there will be more bins in these regions to equalize the number of search points. Figure 2 shows three different variables discretized by a 5-

```
procedure GRS($[\ell, u)$ , $C$)                         ▷ Split interval $[\ell, u)$ into $C$ interval
    max-heap INTERVAL-HEAP                                ▷ Create an empty max-heap [8]
    $N \leftarrow$ population size
    INTERVAL-HEAP.insert($[\ell, u)$, $N$)                ▷ Insert the whole search region with key $N$
    while INTERVAL-HEAP.size() $< C$ do
        $([\ell', u'), N') \leftarrow$ INTERVAL-HEAP.pop()   ▷ Obtain the interval which contains most individuals
        $m \leftarrow$ random$[\ell', u')$                 ▷ Randomly determine the split point
        $N_{\ell'} \leftarrow$ number of individuals in $[\ell', m)$
        $N_{u'} \leftarrow$ number of individuals in $[m, u')$
        INTERVAL-HEAP.insert($[\ell', m)$, $N_{\ell'}$)
        INTERVAL-HEAP.insert($[m, u')$, $N_{u'}$)
    end while
    return INTERVAL-HEAP                                  ▷ Now INTERVAL-HEAP contains $C$ intervals
end procedure
```

Fig. 3 Pseudo code for greedy random split.

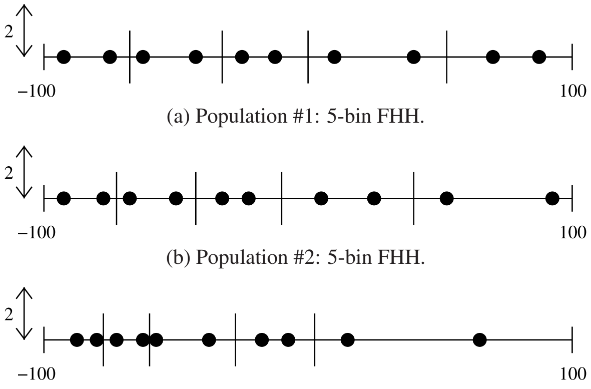


(a) Population #1: 5-bin FWH.

(b) Population #2: 5-bin FWH.

(c) Population #3: 5-bin FWH.

Fig. 1 Three variables discretized by a 5-bin FWH.



(a) Population #1: 5-bin FHH.

(b) Population #2: 5-bin FHH.

(c) Population #3: 5-bin FHH.

Fig. 2 Three variables discretized by a 5-bin FHH.

bin FHH. The fixed height of each bin is 2.

2.2.3 Greedy Random Split

Greedy random split (GRS) is a simple randomized discretization method, of which the main idea is to always randomly split the interval which contains most samples. To implement GRS, we use a heap [8] to store intervals and using the number of individual which is in that interval as key. Thus, we can obtain the interval which contains most samples quickly and then split that interval into two at random. We continue the split step until the demanded number of intervals is reached. The pseudo code of GRS is shown in Fig. 3.

3. Quality of Discretization Methods

In this section, we propose a quality measure of discretization methods and use the proposed measure to analyze the quality of fixed-width histogram, fixed-height histogram, and greedy random split. To the best of our limited knowledge, there is no other similar measure for quantitatively evaluating the quality of discretization methods working with EDAs. Hence, this study can be considered as a start of this line of research, which is definitely worth pursuing.

3.1 Quantitative Evaluation

Because EDAs build probabilistic models based on the selected individuals and discretization methods transform continuous individuals into discrete ones, if a discretization method introduces a lot of distortion into the original population, the probabilistic model built on such a distorted population must be inappropriate for the purpose of model building, and the performance of the whole integrated optimization framework should be greatly reduced. According to this point of view, we propose the use of *distortion distance* between the original population and the discretized population to quantitatively evaluate the quality of discretization methods. The distortion distance is formulated as

$$\int_L^U |p(x) - p^*(x)| \; dx \, , \tag{1}$$

where L and U are the lower bound and the upper bound of a search interval, $p^*(x)$ is the probability density function of the original population distribution, and $p(x)$ is the probability density function of the population distribution discretized by the discretization method to be evaluated. The range of distortion distance calculated with Eq. (1) is interval $[0, 2]$. Distortion distance 0 means the two population distributions are identical, and distortion distance 2 means the two population distributions are disjoint.

We use the defined distortion distance to measure the distortion caused by discretization. Smaller distortion distance means less distortion is introduced to the population. Thus, the resultant (discrete) population should be more similar to the original (continuous) one. According to the scheme of integrating discretization methods into EDAs, if the input for an EDA, i.e., the discrete population, is similar to the original, continuous one, the EDA should be able to accomplish its task. Otherwise, the employed EDA might not be able to handle the problem due to the distortion introduced by discretization. Hence, Eq. (1) is utilized to quantitatively measure the quality of discretization methods.

3.2 Quality Analysis of Fixed-Width Histogram

With the proposed quality measure, we firstly analyze the discretization quality of fixed-width histogram in this section. We will analytically derive the quality of FWH and confirm the derived quality measurement with numerical experiments. Let $N(0, \sigma^2)$ be Gaussian distribution with variance σ^2, $\varphi_{0,\sigma^2}(x)$ be the probability density function, and $\Phi_{0,\sigma^2}(x)$ be the cumulative distribution function. For simplicity, we normalize $N(0, \sigma^2)$ into interval $[-10.0, 10.0)$ to obtain a new distribution $N'(0, \sigma^2)$. The new probability density function is

$$\varphi'_{0,\sigma^2}(x) = \frac{\varphi_{0,\sigma^2}(x)}{\Phi_{0,\sigma^2}(10.0) - \Phi_{0,\sigma^2}(-10.0)} ,$$

and the new cumulative distribution function is

$$\Phi'_{0,\sigma^2}(x) = \frac{\Phi_{0,\sigma^2}(x)}{\Phi_{0,\sigma^2}(10.0) - \Phi_{0,\sigma^2}(-10.0)} .$$

We will use the distribution $N'(0, \sigma^2)$ to model the original, continuous population distribution and examine the quality of discretization methods. The different values of the standard deviation σ, as shown in Fig. 4, are used to model the level of population convergence. Large values of σ are considered as the initial population distribution, in which the individuals are uniformly generated, and small values of σ are considered as the distribution of the population which converges, in which almost all the individuals are close to each other.

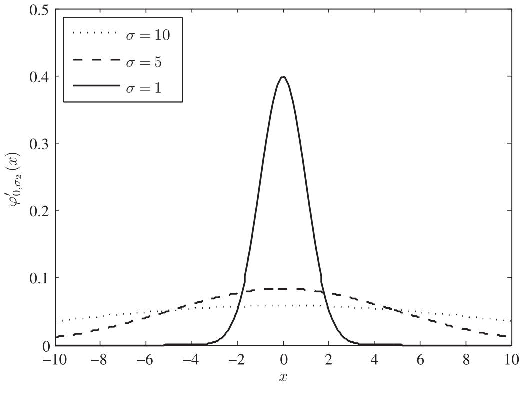


Fig. 4 The probability density function $\varphi'_{0,\sigma^2}(x)$ with σ = 10, 5, and 1.

3.2.1 Analytical Quality Measurement

Firstly, we begin with obtaining the analytical distortion distance of FWH with bin size K, which is assumed to be even without loss of generality, when a population modeled with $N'(0, \sigma^2)$ is discretized by FWH. After discretization, the population distribution created by FWH has K intervals, $I_1, I_2, \cdots, I_K$. The interval corresponds to the ith bin is $I_i = [l_i, u_i)$, where

$$l_i = -10.0 + \frac{20.0 \times (i-1)}{K} ,$$

and

$$u_i = -10.0 + \frac{20.0 \times i}{K} .$$

Since the probability is uniform within each interval, the density of the ith bin is d_i, where

$$\begin{aligned} d_i &= \frac{\Phi'_{0,\sigma^2}(u_i) - \Phi'_{0,\sigma^2}(l_i)}{u_i - l_i} \\ &= \frac{\Phi'_{0,\sigma^2}(u_i) - \Phi'_{0,\sigma^2}(l_i)}{\frac{20.0}{K}} \\ &= \frac{K(\Phi'_{0,\sigma^2}(u_i) - \Phi'_{0,\sigma^2}(l_i))}{20.0} . \end{aligned}$$

Because $N'(0, \sigma^2)$ is symmetric and K is even, we know that in each interval $\varphi'_{0,\sigma^2}(x)$ is monotonically increasing (or decreasing if $i > K/2$.) Hence, the distortion distance can be obtained as

$$\begin{aligned} \sum_{i=1}^{K} \Big(&\left|(m_i - l_i)d_i - (\Phi'_{0,\sigma^2}(m_i) - \Phi'_{0,\sigma^2}(l_i))\right| \\ &+ \left|(u_i - m_i)d_i - (\Phi'_{0,\sigma^2}(h_i) - \Phi'_{0,\sigma^2}(m_i))\right| \Big) , \qquad (2) \end{aligned}$$

where $m_i = \varphi'^{-1}_{0,\sigma^2}(d_i)$. Since there are two solutions of $\varphi'^{-1}_{0,\sigma^2}(d_i)$, we always take the one in $[l_i, u_i)$.

The distortion distance of FWH with 8, 16, and 32 bins are shown in Fig. 5. The x-axis represents the standard de-

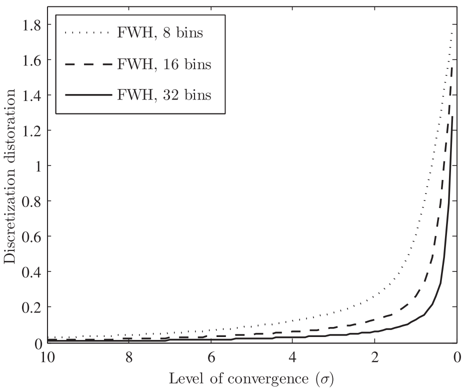


Fig. 5 The distortion distance of FWH obtained analytically. The x-axis is the value of standard deviation, and the y-axis is the distortion distance.

viation varied from 10.0 to 0.1 to model the level of population convergence. From Fig. 5, we can observe that large bin sizes lead to small distortion distances, i.e., less distortion. Such a result agrees with our intuition. When the distribution is almost uniform, the resultant population distribution created by FWH has a very small distortion distance from the original population distribution. However, when the population converges, i.e., $\sigma \to 0$, the distortion distance of FWH becomes very large.

3.2.2 Empirical Quality Measurement

After deriving the analytical quality measurement of FWH, we use a series of numerical experiments to calculate the empirical quality measurements. The goal to obtain both the analytical and empirical quality measurements is two-fold: (1) We would like to confirm the results with approaches of totally different natures, because sometimes analytical results might not be computationally obtained. In order to ensure that the analytical quality measurement derived in this study can be practically realized, numerical sampling procedures should be employed to check if similar quality measurements can be calculated. (2) For advanced, complicated discretization methods, analytical quality measurements might not be easy to derive. In such cases, numerical sampling procedures can always be employed to calculate empirical quality measurements. After being confirmed in the cases of elementary discretization methods, the proposed quality measure may therefore be empirically obtained for advanced, complicated discretization methods.

Figure 6 shows the empirical quality measurements and compares the empirical results to the analytical results. In the series of experiments, different numbers of samples are generated from the distribution $N'(0, \sigma^2)$, which models the population distribution, and taken as the input population of FWH. Numerical sampling procedures are used to calculate the distortion distance between $N'(0, \sigma^2)$ and the distribution of the resultant population created by FWH. We examine the situations involving 1000, 10000, and 100000 samples with bin sizes 8, 16, and 32. The standard deviation is varied from 10.0 to 0.1 with a step of 0.1. For each experimental setting, we perform 50 independently runs and record the mean of calculated distortion distances. As we can observe in Figs. 6 (a) (8 bins), 6 (b) (16 bins), and 6 (c) (32 bins), if the number of samples is sufficiently large, the empirical and analytical results are in a very good agreement.

3.3 Quality Analysis of Fixed-Height Histogram

In this section, the discretization quality of fixed-height histogram is analytically derived and empirically obtained as FWH was in the previous section.

3.3.1 Analytical Quality Measurement

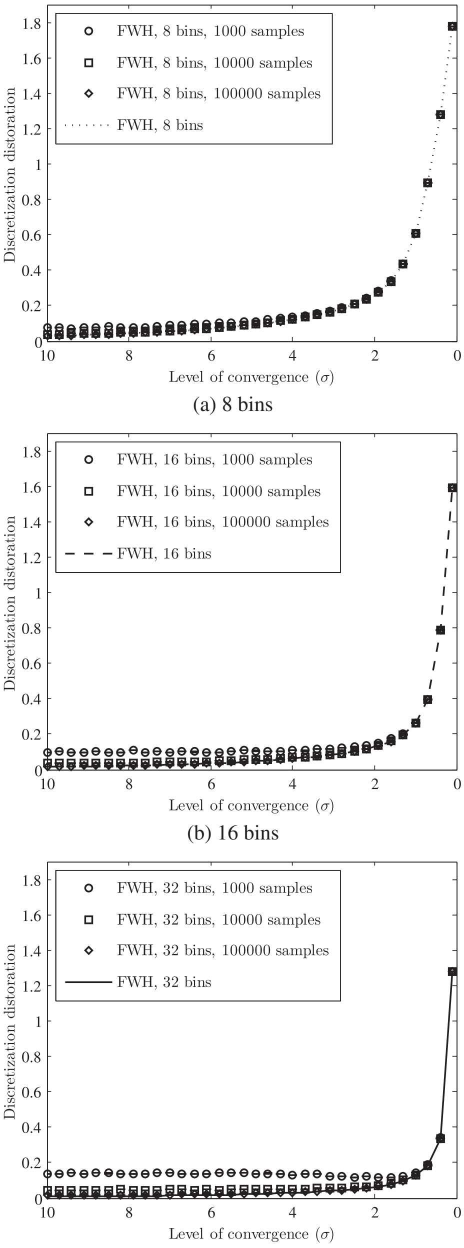
Firstly, we will derive an equation of distortion distance of FHH with bin size K, which is assumed to be even without loss of generality. After discretization, the population distribution created by FHH has K intervals $I_1, I_2, \cdots, I_K$. The interval that corresponds to the ith bin is $I_i = [l_i, u_i)$, where

$$l_i = \Phi'^{-1}_{0,\sigma^2}\left(\frac{i-1}{K}\right), \quad u_i = \Phi'^{-1}_{0,\sigma^2}\left(\frac{i}{K}\right).$$

Since the probability is uniform within each interval, the density of the ith bin is d_i, where

$$d_i = \frac{\Phi'_{0,\sigma^2}(u_i) - \Phi'_{0,\sigma^2}(l_i)}{u_i - l_i} = \frac{\frac{i}{K} - \frac{i-1}{K}}{u_i - l_i} = \frac{1}{K(u_i - l_i)}.$$

Because $N'(0, \sigma^2)$ is symmetric and K is even, in each in-



(a) 8 bins

(b) 16 bins

(c) 32 bins

Fig. 6 The empirical quality measurements obtained with 1000, 10000, and 100000 samples and the analytical quality measurement of FWH.

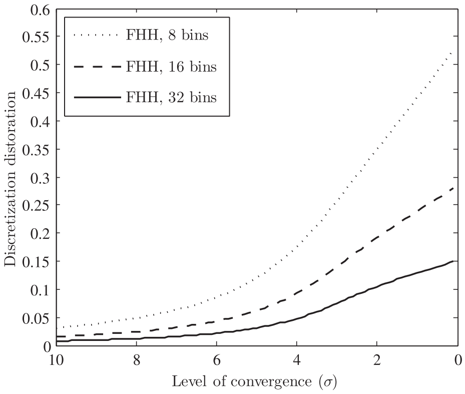


Fig. 7 The distortion distance of FHH obtained analytically. The *x*-axis is the value of standard deviation, and the *y*-axis is the distortion distance.

terval $\varphi'_{0,\sigma^2}(x)$ is monotonically increasing (or decreasing if $i > K/2$). Hence, the distortion distance can be computed as

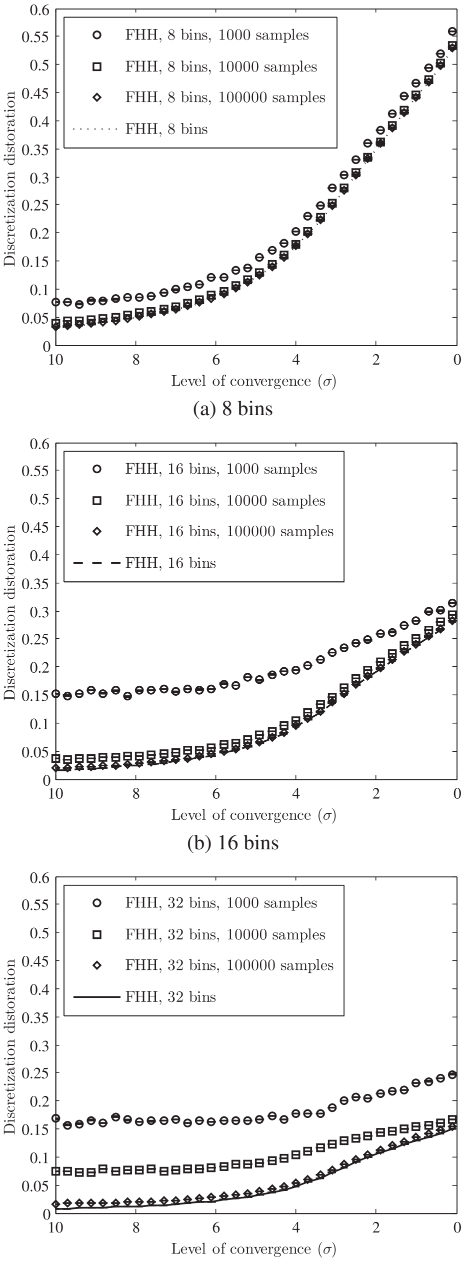
$$\sum_{i=1}^{K}\left(\left|(m_i-l_i)d_i-(\Phi'_{0,\sigma^2}(m_i)-\Phi'_{0,\sigma^2}(l_i))\right| + \left|(u_i-m_i)d_i-(\Phi'_{0,\sigma^2}(h_i)-\Phi'_{0,\sigma^2}(m_i))\right|\right)\ , \qquad (3)$$

where $m_i = \varphi'^{-1}_{0,\sigma^2}(d_i)$, and we also take m_i as the one in $[l_i, u_i)$.

The distortion distance of FHH with bin size 8, 16, and 32 are shown in Fig. 7. The *x*-axis represents the standard deviation, modeling the level of population convergence, from 10 to 0.1. Similar to the analytical result derived for FWH, large bin sizes lead to small distortion distances. However, unlike the result for FWH, when the standard deviation goes close to zero (i.e., the population converges), the distortion distance of FHH grows relatively slowly.

3.3.2 Empirical Quality Measurement

Empirical quality measurements are also obtained for FHH to confirm the consistence of the analytical and sampling approaches. As described in Sect. 3.2.2, the standard deviation is varied from 10.0 to 0.1, and the number of samples are 1000, 10000, and 100000. For each experimental setting, 50 independent runs are performed, and the mean of calculated distortion distances is recorded. The results are shown in Figs. 8 (a) (8 bins), 8 (b) (16 bins), and 8 (c) (32 bins), respectively. If the number of samples is sufficiently large, the empirical and analytical results are also in a very good agreement as we observed on FWH.



(a) 8 bins

(b) 16 bins

(c) 32 bins

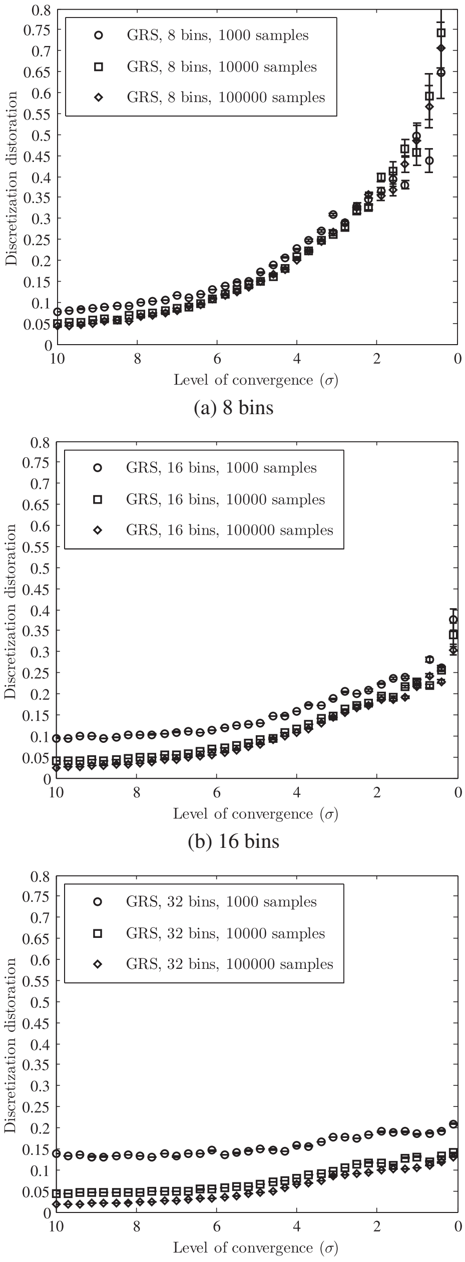
Fig. 8 The empirical quality measurements obtained with 1000, 10000, and 100000 samples and the analytical quality measurement of FHH.

3.4 Quality Analysis of Greedy Random Split

After evaluating the discretization quality of FWH and FHH, we will analyze the discretization quality of greedy random split. The original population distribution is still modeled with $N'(0, \sigma^2)$ which is defined and described in Sect. 3.2. Because of the complexity of deriving the analytical quality measurement for GRS and the confirmed consistence of the analytical and empirical results, we only obtain the empirical quality measurements with similar experiments used in Sects. 3.2.2 and 3.3.2. The standard deviation is varied from 10 to 0.1, the number of samples are 1000, 10000, and 100000, and each experiment is repeated independently for 50 runs. The empirical quality measures for 8, 16, and 32 bins are shown in Fig. 9.

3.5 Discretization Quality Comparison: FWH, FHH, and GRS

According to the quality measurements obtained in previous

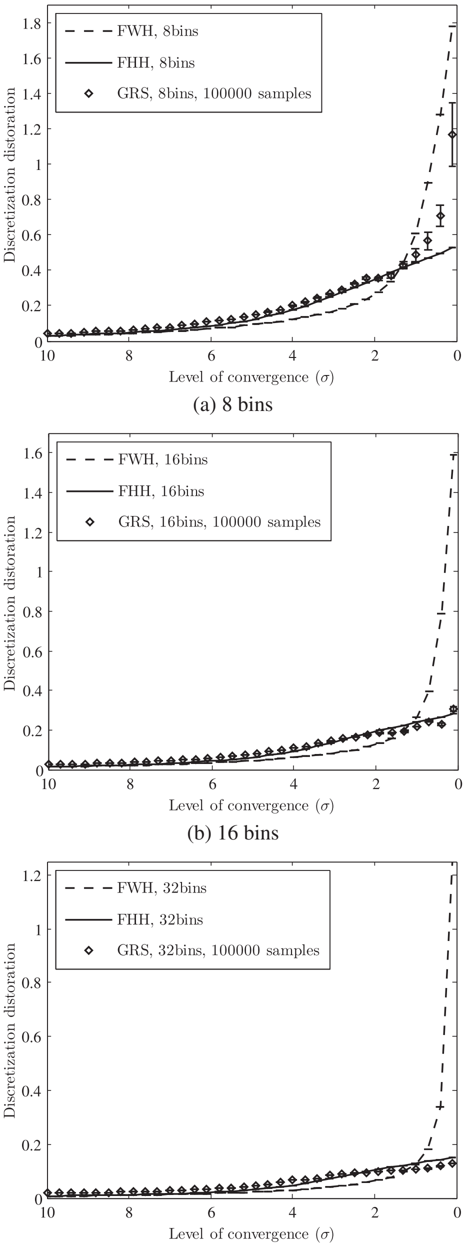


(a) 8 bins

(b) 16 bins

(c) 32 bins

Fig. 9 The empirical quality measurements obtained with 1000, 10000, and 100000 samples of GRS.



(a) 8 bins

(b) 16 bins

(c) 32 bins

Fig. 10 The discretization quality comparison of FWH, FHH, and GRS.

sections, we compare the discretization quality of the three discretization methods: FWH, FHH, and GRS. For simplicity and easiness to observe, we adopt the analytical quality measurements for FWH and FHH and the empirical quality measurements obtained with 100000 samples for GRS. The quality comparison is shown in Fig. 10. More specifically, Figs. 10 (a), 10 (b), and 10 (c) show the distortion distance of FWH, FHH, and GRS with bin sizes 8, 16, and 32, respectively.

In Fig. 10, we can observe that when the standard deviation is large, the distortion distance of FWH is the smallest for the three tested bin sizes, but when the standard deviation decreases (i.e., the population converges,) the distortion distance of FWH becomes the largest in the three methods. For small bin sizes, FHH has the smallest distortion when the standard deviation is small, and when the bin size increased GRS became better than FHH. This comparison suggests that for 8 bins FHH provides the best quality, for 16 bins GRS is slightly better than FHH, and for 32 bins GRS provides the best quality. We will take one step further to examine the three methods working with two different EDAs on benchmark functions in Sect. 4 to reveal the relationship between the proposed quality measure, i.e., the distortion distance, and the performance of EDAs integrated with these discretization methods.

4. Working with EDAs

In this section, we respectively integrate the three investigated discretization methods, FWH, FHH, and GRS with BOA, a representative EDA. We will conduct numerical experiments on test functions to check if the performance benchmarks match the quality measurements. The influence of the adopted discretization methods and the relationship between the proposed quality measure and the performance of different integrations will also be discussed.

4.1 BOA Integrated with Discretization

Bayesian optimization algorithm (BOA) is proposed by [25]. In BOA, a Bayesian network is built from the selected individuals and used to generate offspring. In this study, we use BOA with decision graphs [24], [27]. For more details on BOA, please refer to [25], [27]. BOA can be algorithmically outlined as

```
1. $t \leftarrow 0$, randomly generate initial population $P(0)$
2. Select a set of promising individuals $S(t)$ from $P(t)$
3. Construct network $B$ with a chosen metric and constraints
4. Generate a set of new individuals $O(t)$ according to the joint distribution encoded by $B$
5. Create a new population $P(t+1)$ by replacing some individuals from $P(t)$ with $O(t)$, $t \leftarrow t+1$
6. If the termination criterion is not met, go to step 2
```

After integrated with discretization, the flow becomes

```
1. $t \leftarrow 0$, randomly generate initial population $P(0)$
2. **Conduct discretization to encode each variable of the population.**
3. Select a set of promising individuals $S(t)$ from $P(t)$
4. Construct network $B$ with a chosen metric and constraints
5. Generate a set of new individuals $O(t)$ according to the joint distribution encoded by $B$
6. **Sample the coded intervals to generate the real-valued individuals in $O(t)$.**
7. Create a new population $P(t+1)$ by replacing some individuals from $P(t)$ with $O(t)$, $t \leftarrow t+1$
8. If the termination criterion is not met, go to step 2
```

4.2 Numerical Experiments

We adopt the CEC 2005 benchmark functions [34] to observe the performance difference of BOA integrated with the three discretization methods. The benchmark suite is described briefly in Appendix. For more details, please refer to the original document [34]. The number of dimensions of the benchmark functions is set to 10 to examine FWH, FHH, and GRS with bin sizes 8, 16, and 32, i.e., the discretized problems are 30, 40, and 50 bits.

The parameters of BOA are given in the following. Population size = 400, offspring percentage = 50, tournament size = 8, maximum function evaluations = 30000, and 30 independent runs for each function. The results of BOA are given in Table 1 for each bin size.

In the table, the results of the discretization method that achieves the best performance in each benchmark function are bold-faced, and t-tests are conducted on these results. The results of the other two discretization methods are tested against the best one and are marked by × if outperformed by the best one with a confidence level at least 95%.

4.3 Discussion

According to the experimental results presented in Table 1. BOA with FHH has the best performance when 8 bins are used to encode each variable (outperforming the other two methods on 13 out of 25 functions.) When the bin size increases, the performance of BOA with GRS becomes better compared to the other two methods (outperforming the other two methods on 16 out of 25 functions with 16 bins and on 18 out of 25 functions with 32 bins.)

From Fig. 10 and the discussion in Sect. 3.5, we know that FHH provides the best discretization quality when 8 bins are used. When bin size increases, GRS provides better discretization quality compared to the other two methods. The numerical results of BOA well agree with our discretization quality measure. Given the observed results, we may conclude that there exists a connection between the quality of discretization methods and the performance of EDAs integrated with discretization as well as that the proposed quality measure can reveal such a property and quantitatively assess the discretization methods.

5. Summary and Conclusions

In this paper, we proposed a quality measure of discretization methods. Then, we utilized the proposed quality measure to analyze fixed-width histogram (FWH), fixed-height histogram (FHH), and greedy random split (GRS). Analytical measurements were derived for FWH and FHH, and empirical measurements were obtained for FWH, FHH, and GRS. We compared the three methods with three settings (8, 16, and 32 bins).

In order to study the connection between the quality measure and the performance of EDAs integrated with discretization, we integrated BOA with FWH, FHH, and GRS, respectively, to conduct numerical experiments on test functions. A good agreement between the discretization quality measurements and the numerical optimization results was obtained. As a consequence, this study suggests that there exists a connection between the quality of discretization and the performance of EDAs integrated with discretization as well as that the proposed quality measure provides a quantitative assessment on the suitability for a discretization method to work with EDAs.

The future work along this line includes the use of the proposed quality measure to analyze more discretiza-

Table 1 The numerical results of BOA integrated with FWH, FHH, and GRS (8 bins, 16 bins, and 32 bins).

	FWH		FHH		GRS	
8 bins	mean	var.	mean	var.	mean	var.
F_1	4.140e+01×	4.892e+04	**−4.458e+02**	4.596e+01	−4.362e+02×	1.910e+01
F_2	1.485e+03×	9.210e+05	−2.422e+02×	1.543e+04	**−4.183e+02**	1.305e+02
F_3	2.725e+06×	3.489e+12	2.736e+06×	4.570e+12	**1.166e+06**	3.287e+11
F_4	1.865e+03×	1.147e+06	−1.218e+02×	3.208e+04	**−4.104e+02**	1.798e+02
F_5	2.947e+03×	9.372e+05	1.703e+02×	1.090e+05	**−1.686e+02**	4.880e+02
F_6	4.128e+06×	2.390e+13	**4.602e+03**	2.143e+07	7.852e+03×	1.454e+07
F_7	1.289e+03×	8.658e+03	**1.087e+03**	4.984e−03	1.089e+03×	1.177e−01
F_8	−1.196e+02	1.007e−02	**−1.196e+02**	9.066e−03	−1.196e+02	6.391e−03
F_9	−3.040e+02×	1.950e+01	**−3.252e+02**	2.201e+00	−3.170e+02×	5.651e+00
F_{10}	−2.987e+02×	2.529e+01	**−3.190e+02**	3.191e+01	−2.993e+02×	3.682e+01
F_{11}	9.699e+01×	7.193e−01	**9.488e+01**	9.979e−01	9.498e+01	2.452e+00
F_{12}	4.486e+03×	1.267e+07	1.704e+03×	1.727e+06	**4.667e+02**	6.729e+05
F_{13}	−1.191e+02×	5.076e+01	**−1.289e+02**	2.119e−01	−1.265e+02×	3.273e−01
F_{14}	−2.967e+02	4.775e−02	**−2.968e+02**	7.395e−02	−2.965e+02×	8.050e−02
F_{15}	5.964e+02	1.626e+04	**5.669e+02**	6.629e+03	5.734e+02	1.097e+04
F_{16}	2.955e+02×	1.758e+02	**2.354e+02**	1.012e+02	2.857e+02×	1.474e+02
F_{17}	3.167e+02×	1.511e+02	**2.548e+02**	3.633e+02	3.078e+02×	1.581e+02
F_{18}	9.843e+02×	4.197e+03	8.653e+02×	9.932e+03	**5.707e+02**	5.147e+04
F_{19}	9.705e+02×	3.764e+03	8.559e+02×	1.261e+04	**5.870e+02**	5.180e+04
F_{20}	9.591e+02×	3.809e+03	8.944e+02×	7.990e+03	**6.176e+02**	5.929e+04
F_{21}	1.430e+03×	2.030e+04	1.295e+03×	7.666e+04	**1.168e+03**	6.297e+04
F_{22}	1.223e+03×	1.778e+03	1.170e+03	1.596e+03	**1.158e+03**	7.604e+02
F_{23}	1.460e+03×	1.746e+04	1.274e+03×	9.992e+04	**1.150e+03**	5.733e+04
F_{24}	1.007e+03×	4.140e+04	5.124e+02×	1.564e+03	**4.726e+02**	1.032e+01
F_{25}	2.071e+03×	1.385e+02	**2.021e+03**	1.229e+01	2.026e+03×	5.425e+00
16 bins						
F_1	8.019e+01×	7.939e+04	−4.499e+02×	3.626e−02	**−4.500e+02**	1.112e−05
F_2	1.224e+03×	9.021e+05	−3.830e+02×	1.343e+03	**−4.496e+02**	1.868e−01
F_3	4.177e+06×	2.336e+13	2.069e+06×	2.852e+12	**6.703e+05**	2.986e+11
F_4	1.942e+03×	1.084e+06	−2.801e+02×	2.904e+04	**−4.489e+02**	9.053e−01
F_5	3.161e+03×	8.525e+05	3.454e+01×	7.977e+04	**−3.083e+02**	2.096e−01
F_6	8.723e+06×	6.594e+13	9.012e+02×	3.075e+05	**4.894e+02**	2.558e+04
F_7	1.271e+03×	3.864e+03	1.087e+03×	3.152e−03	**1.087e+03**	5.584e−07
F_8	**−1.196e+02**	7.127e−03	−1.195e+02	8.666e−03	−1.196e+02	9.111e−03
F_9	−3.105e+02×	1.776e+01	**−3.282e+02**	1.598e+00	−3.266e+02×	1.062e+00
F_{10}	−3.023e+02×	2.533e+01	−3.224e+02	1.001e+01	**−3.227e+02**	3.511e+01
F_{11}	9.753e+01×	4.680e−01	9.428e+01	1.221e+00	**9.375e+01**	2.300e+00
F_{12}	5.210e+03×	7.609e+06	7.779e+02×	8.125e+05	**2.978e+02**	1.117e+06
F_{13}	−1.156e+02×	1.092e+02	**−1.293e+02**	3.590e−02	−1.291e+02×	1.127e−01
F_{14}	−2.968e+02×	7.884e−02	**−2.969e+02**	1.296e−01	−2.969e+02	9.851e−02
F_{15}	5.303e+02×	1.684e+04	**4.645e+02**	2.013e+04	5.034e+02	8.692e+03
F_{16}	2.805e+02×	3.667e+02	**2.270e+02**	5.781e+01	2.275e+02	1.745e+02
F_{17}	3.017e+02×	2.422e+02	2.367e+02×	1.620e+02	**2.304e+02**	1.435e+02
F_{18}	9.909e+02×	4.028e+03	8.209e+02	1.495e+04	**8.073e+02**	3.783e+04
F_{19}	9.526e+02×	6.789e+03	8.285e+02×	6.825e+03	**7.367e+02**	4.554e+04
F_{20}	9.829e+02×	2.609e+03	8.555e+02×	1.024e+04	**7.691e+02**	3.898e+04
F_{21}	1.446e+03×	1.365e+04	**1.164e+03**	6.878e+04	1.212e+03	4.275e+04
F_{22}	1.227e+03×	1.828e+03	**1.137e+03**	5.229e+02	1.140e+03	5.760e+02
F_{23}	1.522e+03×	4.424e+03	**1.290e+03**	9.517e+04	1.374e+03	5.098e+04
F_{24}	1.042e+03×	4.046e+04	4.604e+02×	1.257e−01	**4.600e+02**	1.334e−05
F_{25}	2.067e+03×	1.867e+02	2.022e+03×	3.219e+01	**2.018e+03**	2.588e+01
32 bints						
F_1	1.437e+02×	1.143e+05	−4.500e+02×	2.132e−03	**−4.500e+02**	0.000e+00
F_2	1.512e+03×	1.025e+06	−3.769e+02×	5.531e+03	**−4.457e+02**	1.749e+01
F_3	4.558e+06×	1.089e+13	2.043e+06×	3.584e+12	**1.062e+06**	1.069e+12
F_4	2.019e+03×	1.811e+06	−2.882e+02×	2.384e+04	**−4.070e+02**	3.421e+03
F_5	3.101e+03×	9.279e+05	2.204e+02×	1.098e+05	**−1.946e+02**	5.986e+04
F_6	1.579e+07×	5.365e+14	1.270e+03×	1.668e+06	**4.919e+02**	5.131e+04
F_7	1.287e+03×	5.079e+03	1.087e+03×	5.063e−03	**1.087e+03**	1.387e−09
F_8	**−1.196e+02**	7.520e−03	−1.195e+02	8.492e−03	−1.195e+02	5.271e−03
F_9	−3.143e+02×	1.365e+01	**−3.291e+02**	7.497e−01	−3.280e+02×	1.708e+00
F_{10}	−3.033e+02×	2.456e+01	−3.223e+02×	1.052e+01	**−3.236e+02**	4.463e+00
F_{11}	9.749e+01×	8.084e−01	9.488e+01	1.940e+00	**9.479e+01**	3.229e+00
F_{12}	7.516e+03×	1.152e+07	9.071e+02×	1.617e+06	**1.183e+02**	4.071e+05
F_{13}	−1.198e+02×	5.411e+01	**−1.293e+02**	5.409e−02	−1.293e+02	8.581e−02
F_{14}	−2.968e+02×	1.357e−01	**−2.970e+02**	1.495e−01	−2.969e+02	1.730e−01
F_{15}	5.269e+02×	2.018e+04	4.309e+02	2.868e+04	**4.045e+02**	2.800e+04
F_{16}	2.881e+02×	5.819e+02	**2.290e+02**	6.953e+01	2.291e+02	7.620e+01
F_{17}	2.962e+02×	5.607e+02	**2.385e+02**	1.436e+02	2.412e+02	9.208e+02
F_{18}	9.971e+02×	2.531e+03	8.659e+02	9.969e+03	**8.531e+02**	2.179e+04
F_{19}	9.952e+02×	1.715e+03	8.490e+02	6.753e+03	**8.432e+02**	1.334e+04
F_{20}	9.981e+02×	3.040e+03	8.373e+02	1.465e+04	**8.289e+02**	1.847e+04
F_{21}	1.487e+03×	1.727e+04	1.147e+03	6.684e+04	**1.088e+03**	5.931e+04
F_{22}	1.224e+03×	2.484e+03	1.139e+03	8.854e+02	**1.137e+03**	3.026e+02
F_{23}	1.573e+03×	1.202e+04	**1.227e+03**	9.622e+04	1.427e+03×	3.688e+04
F_{24}	1.069e+03×	2.939e+04	4.602e+02×	6.249e−02	**4.600e+02**	0.000e+00
F_{25}	2.065e+03×	1.249e+02	2.026e+03×	4.610e+01	**2.022e+03**	1.869e+01

tion methods. Since it has been verified in this study that based on the proposed quality measure, the analytical measurement and the empirical measurement are consistent, for complicated discretization methods, numerical sampling techniques can be adopted to evaluate the discretization quality. Moreover, how to set the bin size should also be carefully studied, since a larger bin size means a larger discretized problem for EDAs. A large bin size, e.g., close to the population size, may greatly influence the algorithmic performance. The resultant probabilistic model will most likely overfit, and hence, a proper ratio should be maintained between the adopted bin size and the population size. Finally, the connection between the discretization quality and the optimization performance should be further investigated. Essential insights and important understandings of the EDA working principles might be revealed.

Acknowledgments

The work was supported in part by the National Science Council of Taiwan under Grant NSC 101-2628-E-009-024-MY3. The authors are grateful to the National Center for High-performance Computing for computer time and facilities.

References

[1] C.W. Ahn, R.S. Ramakrishna, and D.E. Goldberg, “Real-coded Bayesian optimization algorithm, bringing the strength of BOA into the continuous world,” Proc. Genetic and Evolutionary Computation Conference (GECCO-2004), pp.840–851, 2004.

[2] S. Baluja, “Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,” Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1994.

[3] S. Baluja and S. Davies, “Using optimal dependency-trees for combinational optimization,” Proc. Fourteenth International Conference on Machine Learning, pp.30–38, 1997.

[4] P.A. Bosman and J. Grahl, “Matching inductive search bias and problem structure in continuous estimation of distribution algorithms,” European Journal of Operational Research, vol.185, pp.1246–1264, 2008.

[5] P.A. Bosman and D. Thierens, “Numerical optimization with real-valued estimation-of-distribution algorithms,” in Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications, ed. M. Pelikan, K. Sastry, and E. Cantú-Paz, Studies in Computational Intelligence, pp.91–120. Springer, 2006.

[6] C.-H. Chen, W.-N. Liu, and Y.-P. Chen, “Adaptive discretization for probabilistic model building genetic algorithms,” Proc. ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2006), pp.1103–1110, 2006.

[7] Y.-P. Chen and C.-H. Chen, “Enabling the extended compact genetic algorithm for real-parameter optimization by using adaptive discretization,” Evolutionary Computation, vol.18, no.2, pp.199–228, 2010.

[8] T.H. Cormen, C. Stein, R.L. Rivest, and C.E. Leiserson, Introduction to Algorithms, 2nd ed., McGraw-Hill Higher Education, 2001.

[9] J. de Bonet, C. Isbell, and P. Viola, “MIMIC: Finding optima by estimating probability densities,” Advances in Neural Information Processing Systems, vol.9, pp.424–430, 1997.

[10] R. Etxeberria and P. Larrañaga, “Global optimization using Bayesian networks,” Proc. Second Symposium on Artificial Intelligence, pp.332–339, 1999.

[11] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, New York, 1989.

[12] D.E. Goldberg, The Design of Innovation: Lessons from and for Competent Genetic Algorithms, volume 7 of Genetic Algorithms and Evoluationary Computation, Kluwer Academic Publishers, 2002. ISBN: 1-4020-7098-5.

[13] G.R. Harik, “Linkage learning via probabilistic modeling in the ECGA,” IlliGAL Report No.99010, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.

[14] G.R. Harik, F.G. Lobo, and D.E. Goldberg, “The compact genetic algorithm,” IEEE Trans. Evolution. Computation, vol.3, no.4, pp.287–297, Nov. 1999.

[15] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975. ISBN: 0-262-58111-6.

[16] S. Kern, S.D. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos, “Learning probability distributions in continuous evolutionary algorithms – A comparative review,” Natural Comput., vol.3, no.1, pp.77–112, 2004.

[17] S. Kotsiantis and D. Kanellopoulos, “Discretization techniques: A recent survey,” GESTS International Transactions on Computer Science and Engineering, vol.32, no.1, pp.47–58, 2006.

[18] P. Lanzi, L. Nichetti, K. Sastry, D. Voltini, and D. Goldberg, “Real-coded extended compact genetic algorithm based on mixtures of models,” in Linkage in Evolutionary Computation, ed. Y.-P. Chen and M.-H. Lim, volume 157 of Studies in Computational Intelligence, chapter 14, pp.335–358, Springer, Berlin, Heidelberg, 2008.

[19] P. Larrañaga and J.A. Lozano, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, volume 2 of Genetic algorithms and evolutionary computation, Kluwer Academic Publishers, Boston, MA, 2001. ISBN: 0-7923-7466-5.

[20] J.A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing), Springer-Verlag, New York, Secaucus, NJ, USA, 2006.

[21] H. Mühlenbein and R. Höns, “The estimation of distributions and the minimum relative entropy principle,” Evolutionary Computation, vol.13, no.1, pp.1–27, 2005.

[22] H. Mühlenbein and T. Mahnig, “FDA: A scalable evolutionary algorithm for the optimization of additively decomposed functions,” Evolutionary Computation, vol.7, no.4, pp.353–376, 1999.

[23] H. Mühlenbein and G. Paaß, “From recombination of genes to the estimation of distributions I. binary parameters,” Proc. 4th Conference on Parallel Problem Solving from Nature (PPSN IV), pp.178–187, 1996.

[24] M. Pelikan, “A simple implementation of the Bayesian optimization algorithm (BOA) in C++ (version 1.0),” IlliGAL Report No.99011, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.

[25] M. Pelikan, D.E. Goldberg, and E. Cantú-Paz, “BOA: The Bayesian optimization algorithm,” Proc. Genetic and Evolutionary Computation Conference (GECCO-99), pp.525–532, 1999.

[26] M. Pelikan, D.E. Goldberg, and F.G. Lobo, “A survey of optimization by building and using probabilistic models,” Computational Optimization and Applications, vol.21, no.1, pp.5–20, 2002.

[27] M. Pelikan, D.E. Goldberg, and K. Sastry, “Bayesian optimization algorithm, decision graphs, and Occam’s razor,” Proc. Genetic and Evolutionary Computation Conference (GECCO-2001), pp.519–526, 2001.

[28] M. Pelikan, D.E. Goldberg, and S. Tsutsui, “Getting the best of both worlds: Discrete and continuous genetic and evolutionary algorithms in concert,” Information Sciences, vol.156, pp.147–171, 2003.

[29] M. Pelikan and H. Mühlenbein, “The bivariate marginal distribution algorithm,” Advances in Soft Computing, pp.521–535, 1999.

[30] M. Pelikan, K. Sastry, and D.E. Goldberg, “iBOA: The incremental

Bayesian optimization algorithm," GECCO '08: Proc. 10th Annual Conference on Genetic and Evolutionary Computation, pp.455–462, New York, NY, USA, ACM, 2008.

[31] M. Sebag and A. Ducoulombier, "Extending population-based incremental learning to continuous search spaces," Proc. Fifth International Conference on Parallel Problem Solving from Nature (PPSN V), pp.418–427, 1998.

[32] I.L. Servet, L. Trave-Massuyes, and D. Stern, "Telephone network traffic overloading diagnosis and evolutionary computation techniques," Proc. Third European Conference on Artificial Evolution (AE 97), pp.137–144, 1997.

[33] S.-Y. Shin and B.-T. Zhang, "Bayesian evolutionary algorithms for continuous function optimization," Proc. 2001 Congress on Evolutionary Computation (CEC2001), pp.508–515, 2001.

[34] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Technical report, Nanyang Technological University, Singapore, May 2005.

[35] S. Tsutsui, M. Pelikan, and D.E. Goldberg, "Evolutionary algorithm using marginal histogram models in continuous domain," IlliGAL Report No.2001019, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2001.

Appendix: CEC 2005 Real-Parameter Optimization Benchmark Suite

The benchmark consists of 5 unimodal functions, 9 multimodal functions, and 11 composition functions. Because the last 13 functions are composed by the first 12 functions, we briefly describe the first 12 functions. For more details, including all the predefined values, please refer to the original document [34]. In the following definition, D denotes the number of dimension. $x = [x_1, x_2, \ldots, x_D]$ is the input, a D-dimensional vector. $N(0,1)$ is a random sample from the normal distribution with mean 0 and variance 1 to create noise. M_i is a predefined linear transform matrix. $o_i = [o_1, o_2, \ldots, o_D]$ is a predefined shift vector, and f_bias_i is a predefined bias.

$$F_1(x) = \sum_{i=1}^{D} z_i^2 + f\_bias_1, z = x - o_1 \tag{A· 1}$$

$$F_2(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 + f\_bias_2, z = x - o_2 \tag{A· 2}$$

$$F_3(x) = \sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}} z_i^2 + f\_bias_3, z = x - o_3 \tag{A· 3}$$

$$\begin{aligned} F_4(x) = &\left(\sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2\right) \times (1 + 0.4|N(0,1)|) \\ &+ f\_bias_4\ , \\ & z = x - o_4 \end{aligned} \tag{A· 4}$$

$$\begin{aligned} &F_5(x) = \max\{A_i x - B_i\} + f\_bias_5\ , \\ &A\text{, a } D \times D \text{ matrix, and } B\text{, a } D \times 1 \text{ vector,} \\ &\text{are predefined.} \end{aligned} \tag{A· 5}$$

$$\begin{aligned} F_6(x) = &\sum_{i=1}^{D-1}(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) \\ &+ f\_bias_6\ , \\ & z = x - o_6 + 1 \end{aligned} \tag{A· 6}$$

$$\begin{aligned} &F_7(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f\_bias_7\ , \\ &z = (x - o_7) \times M_7 \end{aligned} \tag{A· 7}$$

$$\begin{aligned} F_8(x) = &-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) \\ &- \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) \\ &+ 20 + e + f\_bias_8\ , z = (x - o_8) \times M_8 \end{aligned} \tag{A· 8}$$

$$\begin{aligned} &F_9(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f\_bias_9\ , \\ &z = x - o_9 \end{aligned} \tag{A· 9}$$

$$\begin{aligned} &F_{10}(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f\_bias_{10}\ , \\ &z = (x - o_{10}) \times M_{10} \end{aligned} \tag{A· 10}$$

$$\begin{aligned} F_{11}(x) = &\sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}(a^k \cos(2\pi b^k(z_i + 0.5)))\right) \\ &- D\sum_{k=0}^{kmax}(a^k \cos(2\pi b^k \cdot 0.5)) + f\_bias_{11}, \\ & z = (x - o_{11}) \times M_{11}, a = 0.5, b = 3, kmax = 20 \end{aligned} \tag{A· 11}$$

$$\begin{aligned} &F_{12}(x) = \sum_{i=1}^{D}(A_i - B_i(x))^2 + f\_bias_{12}, \\ &z = x - o_{12}\ , \end{aligned} \tag{A· 12}$$

A and B are predefined $D \times D$ matrixes

Chao-Hong Chen received the B.S. and the M.S. degree in Department of Computer Science from National Chiao Tung University, Taiwan, in 2007 and 2009, and currently studies for the Ph.D. degree in Computer Science at National Chiao Tung University, Taiwan.

Ying-ping Chen is currently an Associate Professor in the Department of Computer Science, National Chiao Tung University, Taiwan. His research interests include data grid and MapReduce technologies in distributed computation as well as theories, working principles, and dimensional/facet-wise models in genetic and evolutionary computation. He received the B.S. degree and the M.S. degree in Computer Science and Information Engineering from National Taiwan University, Taiwan, in 1995 and 1997, respectively, and the Ph.D. degree in 2004 from the Department of Computer Science, University of Illinois at Urbana-Champaign, Illinois, USA.